Bundesamt
für Sicherheit in der
Informationstechnik

Technical Guideline

Advanced Security Mechanisms for Machine Readable Travel Documents –
Extended Access Control (EAC), Version 1.0

TR-03110

Bundesamt für Sicherheit in der Informationstechnik

Godesberger Allee 185-189, 53175 Bonn ▪ Postfach 200363, 53133 Bonn
Tel.: +49 (0) 1888 9582-0 ▪ Fax: +49 (0) 1888 9582-400 ▪ Internet: www.bsi.bund.de

Contact: epass@bsi.bund.de

Contents

List of Figures

List of Tables

1. Introduction

The next generation of machine readable travel documents (MRTDs) will be equipped with a contactless RF-chip containing digitized biometrics of the holder. Given the nature of digital data, one can easily see that the authenticity (including integrity), originality, and confidentiality of the data stored on the MRTD chip must be appropriately protected. ICAO [6] has therefore specified *Passive Authentication*, *Active Authentication*, and *Access Control* as summarized in Table 1.1.

Table 1.1.: ICAO Security Mechanisms

Mechanism	**Protection**	**Cryptographic Technique**
Passive Authentication	Authenticity	Digital Signature
Active Authentication	Originality	Challenge-Response
Access Control	Confidentiality	Authentication & Secure Channels

While the implementation of Passive Authentication is mandatory, Active Authentication and Access Control are both optional. It directly follows that without implementing those or equivalent mechanisms the originality and confidentiality of the stored data cannot be guaranteed. Active Authentication and Access Control are therefore two important ingredients for a secure MRTD.

1.1. Active Authentication

Cloning MRTD chips is a serious attack. The goal of this attack is to copy the data of an (almost) biometric twin to the attacker's MRTD chip embedded in a genuine passport book. Especially if only the facial image is used as biometric, finding a biometric twin is possible by skimming the data of a few thousands of MRTD chips.

However, even with Passive Authentication, one can detect cloned MRTD chips by carefully comparing both the picture and the machine readable zone (MRZ) printed on the datapage to the data stored on the MRTD chip. This test however only relies on the physical security features of the datapage.

Active Authentication prevents cloning by introducing a chip-individual key pair. The MRTD chip's public key is contained in the signed data, but the private key is stored in secure memory and cannot be copied. The chip can however prove knowledge of its private key in a challenge-response protocol, which is called Active Authentication. In this protocol the MRTD chip digitally signs a challenge randomly chosen by the inspection system. The inspection system is convinced that the MRTD chip is genuine if and only if the returned signature is correct. Active Authentication is a straightforward protocol and prevents cloning very effectively, but introduces a privacy threat: Challenge Semantics (see Appendix D for a discussion on Challenge Semantics).

1.2. Access Control

Access Control is not only required for privacy reasons, but also makes cloning attacks more difficult. Searching for biometric twins is obviously made more difficult by keeping the biometrics as confidential as possible. Access Control comes in two flavors depending on the sensitivity level of the data to be protected:

- Less-sensitive data (e.g. the MRZ, the facial image and other data that is relatively easy to acquire from other sources) required for global interoperable border crossing is protected by *Basic Access Control*. For the reader's convenience, Basic Access Control is sketched in Appendix C.

- Sensitive data (e.g. fingerprints and other data that cannot be obtained easily from other sources at a large scale) must only be available to authorized inspection systems. Such data is protected by *Extended Access Control*.

Basic Access Control only checks that the reader has physical access to the passport by requiring the MRZ to be read optically. Extended Access Control should additionally check that the inspection system is entitled to read sensitive data. Therefore, strong authentication of the inspection system is required. However, as Extended Access Control is not required for global interoperable border crossing, this protocol is not (yet) specified by ICAO.

1.3. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [2].

1.4. Abbreviations

Name	Abbrev.
Card Verifiable	CV
Chip Authentication	CA
Chip Authentication Public Key	PK_{ICC}
Chip Authentication Private Key	SK_{ICC}
Country Signing CA	CSCA
Country Verifying CA	CVCA
Country Verifying CA Certificate	C_{CVCA}
Document Security Object	SO_D
Document Verifier	DV
Document Verifier Certificate	C_{DV}
Domain Parameters	$\mathcal{D}$
Integrated Circuit Chip	ICC
Interface Device	IFD
International Civil Aviation Organization	ICAO
Inspection System	IS
Inspection System Certificate	C_{IS}
Key Derivation Function	KDF
Logical Data Structure	LDS
Machine Readable Travel Document	MRTD
Terminal Authentication	TA
Terminal Authentication Public Key	PK_{IFD}
Terminal Authentication Private Key	SK_{IFD}

2. Advanced Security Mechanisms

This document specifies two advanced security mechanisms for machine readable travel documents: *Chip Authentication* and *Terminal Authentication*. While Chip Authentication can be used as a stand-alone protocol e.g. to replace Active Authentication, Terminal Authentication can only be used in combination with Chip Authentication. Together both protocols provide an implementation of Extended Access Control.

Chip Authentication

This protocol is an alternative to the optional Active Authentication Protocol, i.e. it allows the inspection system to verify that the MRTD chip is genuine but has two advantages over the original protocol.

- Challenge Semantics are prevented because the transcripts produced by this protocol are non-transferable.
- Besides authentication of the MRTD chip this protocol also provides strong session encryption.

An MRTD chip that supports Chip Authentication MUST also enforce Basic Access Control.

Terminal Authentication

This protocol allows the MRTD chip to verify that the inspection system is entitled to access sensitive data. As the inspection system MAY access sensitive data afterwards, all further communication MUST be protected appropriately. Therefore, the Chip Authentication Protocol MUST have been successfully executed before starting this protocol – which is enforced by the protocol itself.

2.1. Inspection Procedure

Depending on whether or not a device (i.e. an MRTD chip or an inspection system) is compliant to this specification the device is called *compliant* or *non-compliant*, respectively. Depending on the combination of an inspection system and an MRTD chip, either a *standard inspection procedure* or an *advanced inspection procedure* is used:

- A non-compliant inspection system uses the standard inspection procedure. The less-sensitive data stored on a compliant MRTD chip MUST be readable by every non-compliant inspection system.
- A compliant inspection system MUST use the appropriate inspection procedure, i.e.

Table 2.1.: Inspection Procedures

Inspection system	**MRTD chip**	
	compliant	**non-compliant**
compliant	Advanced	Standard
non-compliant	Standard	Standard

- the advanced inspection procedure for compliant MRTD chips and
- the standard inspection procedure for non-compliant MRTD chips.

Table 2.1 gives an overview on the inspection procedures to be used.

2.1.1. Standard Inspection Procedure

The standard inspection procedure consists of the following steps:

1. Basic Access Control (Conditional)
2. Passive Authentication (OPTIONAL)
3. Reading of less-sensitive data (OPTIONAL)

If Basic Access Control is enforced by the MRTD chip, this step MUST be performed prior to all other steps. If the MRTD chip does not enforce Basic Access Control, this step MUST NOT be used. The order of the remaining steps is irrelevant.

2.1.2. Advanced Inspection Procedure

The advanced inspection procedure consists of the following steps:

1. Basic Access Control (REQUIRED)
2. Chip Authentication (REQUIRED)
3. Passive Authentication (REQUIRED)
4. Reading of less-sensitive data (OPTIONAL)
5. Terminal Authentication (OPTIONAL)
6. Reading of sensitive data (OPTIONAL)

Mandatory steps MUST be performed in the order indicated. The order of optional steps is irrelevant, however, Terminal Authentication MUST be successfully performed before sensitive data can be read. This MUST be enforced by the MRTD chip.

Note: After a successful execution of Chip Authentication strong session encryption is established rendering the decryption of an eavesdropped communication computationally impossible.

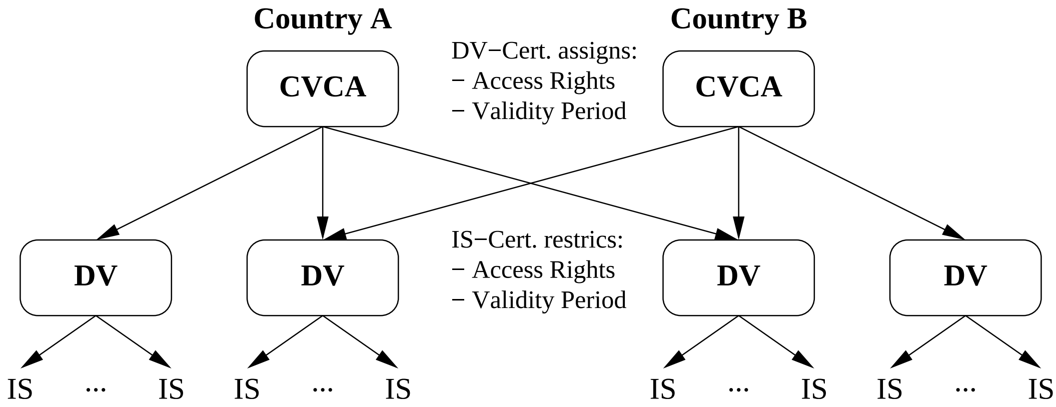

Figure 2.1.: Extended Access Control PKI

2.2. Public Key Infrastructure

Terminal Authentication requires the inspection system to prove that it is entitled to access sensitive data. Such an inspection system is equipped with at least one *Inspection System Certificate,* encoding the inspection system's public key and access rights, and the corresponding private key. After the inspection system has proven knowledge of this private key, the MRTD chip grants the inspection system access to sensitive data as indicated in the Inspection System Certificate.

The PKI required for issuing and validating Inspection System Certificates consists of the following entities:

1. Country Verifying CAs issuing Country Verifying CA Link-Certificates and Document Verifier Certificates.
2. Document Verifiers issuing Inspection System Certificates.
3. Inspection systems accessing MRTD chips.

This PKI forms the basis of Extended Access Control. It is illustrated in Figure 2.1.

2.2.1. Country Verifying CA

Every State is required to set up one trustpoint that issues Document Verifier Certificates: the *Country Verifying CA* (CVCA).

Note: The Country Signing CA issuing Certificates for Document Signers (cf. [6]) and the Country Verifying CA MAY be integrated into a single entity, e.g. a Country CA. However, even in this case, separate key pairs MUST be used for different roles.

A CVCA determines the access rights to "its" MRTD chips for all DVs (i.e. its own DVs as well as the DVs of other States) by issuing certificates for DVs entitled to access some sensitive data. The conditions under which a CVCA grants a DV access to sensitive data is out of the scope of this document. It is however RECOMMENDED that those conditions are stated in a certificate policy published by the CVCA.

Document Verifier Certificates MUST contain information, which data a certain DV is entitled to access. To diminish the potential risk introduced by lost or stolen inspection systems Document Verifier Certificates MUST contain a short validity period. The validity period is assigned by the issuing CVCA at its own choice and may differ depending on the Document Verifier the certificate is issued for.

2.2.2. Document Verifiers

A *Document Verifier* (DV) is an organizational unit that manages inspection systems belonging together (e.g. inspection systems operated by a State's border police) by – inter alia – issuing Inspection System Certificates. A Document Verifier is therefore a CA, authorized by the national CVCA to issue certificates for national inspection systems. The Inspection System Certificates issued by a DV usually inherit both the access rights and the validity period from the Document Verifier Certificate, however, the Document Verifier MAY choose to further **restrict** the access rights or the validity period depending on the inspection system the certificate is issued for.

If a Document Verifier requires its inspection systems to access sensitive data stored on other States' MRTD chips, it MUST send a Certification Request (containing its Public key) to the respective State's CVCA in order to receive the required Document Verifier Certificate.

The Document Verifier MUST also ensure that all received Document Verifier Certificates are forwarded to the inspection systems.

2.2.3. Card Verifiable Certificates

CVCA Certificates, DV Certificates, and IS Certificates are to be validated by MRTD chips. Due to the computational restrictions of those chips, the certificates MUST be in a card verifiable format:

- The certificate format and profile is specified in Appendix A.3.1.
- The signature algorithm, domain parameters, and key sizes to be used are determined by the CVCA of the issuing State, i.e. the same signature algorithm, domain parameters and key sizes MUST be used within a certificate chain.[1]
- CVCA link certificates MAY include a public key that deviates from the current parameters, i.e. the CVCA MAY switch to a new signature algorithm, new domain parameters, or key sizes.

2.2.3.1. Certificate Scheduling

Each certificate MUST contain a validity period. This validity period is identified by two dates, the *certificate effective date* and the *certificate expiration date.*

[1] As a consequence Document Verifiers and inspection systems will have to be provided with several key pairs.

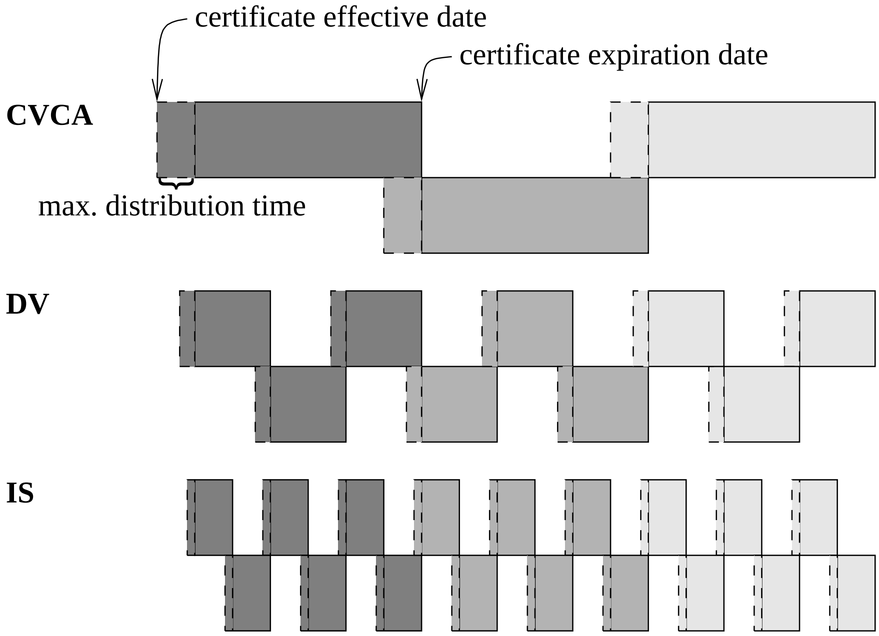

Figure 2.2.: Certificate Scheduling

Certificate Effective Date: The certificate effective date SHALL be the date of the certificate generation.

Certificate Expiration Date: The certificate expiration date MAY be arbitrarily chosen by the certificate issuer.

When generating certificates the issuer MUST carefully plan the renewal of certificates, as sufficient time for propagation of certificates and set up of certificate chains MUST be provided. Obviously, a new certificate must be generated before the certificate to be replaced expires. The resulting *maximum distribution time* equals the certificate expiration date of the old certificate minus the certificate effective date of the new certificate. Certificate scheduling is illustrated in Figure 2.2.

2.2.3.2. Certificate Distribution

For distribution of CVCA and DV Certificates between States the communication channels specified in Appendix A.4 SHALL be used. The distribution of IS Certificates and the propagation of CVCA and DV Certificates within a State is out of the scope of this specification.

2.2.4. Certificate Validation

To validate an IS Certificate, the MRTD chip must be provided with a certificate chain starting at the MRTD chip's trustpoint. This trustpoint is a more or less recent public key of the MRTD chip's CVCA. The initial trustpoint is stored in the MRTD chip's secure memory in the production or (pre-) personalization phase.

As the key pair used by the CVCA changes over time, CVCA link certificates have to be produced. The MRTD chip is REQUIRED to internally update its trustpoint according to received valid link certificates.

The MRTD chip MUST only accept *recent* IS Certificates. As the MRTD chip has no internal clock, the *current date* is approximated as described below. Thus, the MRTD chip only verifies that a certificate is *apparently* recent (i.e. with respect to the approximated current date).

Current Date: The current date stored on the MRTD chip is initially the date of the (pre-) personalization. This date is then autonomously approximated by the MRTD chip using the most recent certificate effective date contained in a valid CVCA Link Certificate, a DV Certificate or a domestic IS Certificate.

The following validation procedure MAY be used to validate a certificate chain. For each received certificate the MRTD chip performs the following steps:

1. The MRTD chip verifies the signature on the certificate. If the signature is incorrect, the verification fails.

2. The certificate expiration date is compared to the MRTD chip's current date. If the expiration date is before the current date, the verification fails.

3. The certificate is valid and the public key and the relevant attributes contained in the certificate are imported.

    a) For CVCA Certificates:
    The new CVCA public key is added to the list of trustpoints stored in the MRTD chip's secure memory. The new trustpoint is then *enabled*.

    b) For DV and IS Certificates:
    The new DV or IS public key is temporarily imported for subsequent certificate verification respectively Terminal Authentication.

4. For CVCA, DV, and domestic IS Certificates:
    The certificate effective date is compared to the MRTD chip's current date. If the current date is before the effective date, the current date is *updated* to the effective date.

5. Expired trustpoints stored in the MRTD chip's secure memory are *disabled* and may be removed from the list of trustpoints.

The operations for *enabling* or *disabling* a CVCA public key and the operation for *updating* the current date MUST be implemented as atomic operation.

Note: Due to the scheduling of CVCA certificates (cf. Figure 2.2), at most two trustpoints need to be stored on the MRTD chip.

3. Protocol Specifications

In this section cryptographic protocols for Chip Authentication and Terminal Authentication are specified assuming an arbitrary communication infrastructure. A mapping to ISO 7816 commands is given in Appendix B.

3.1. Cryptographic Algorithms and Notation

The protocols are executed between two parties: the MRTD chip (ICC) and the inspection system (IFD). The following cryptographic operations and notations are used.

3.1.1. Key Agreement

The keys and operations for key agreement are described in an algorithm-independent way. A mapping to DH and ECDH can be found in Appendix A.1.

3.1.1.1. Keys

- The MRTD chip has a static Diffie-Hellman key pair (or Chip Authentication Key Pair). The public key is PK_{ICC}, the corresponding private key is SK_{ICC}, the domain parameters are $\mathcal{D}_{ICC}$.
- The inspection system generates an ephemeral Diffie-Hellman key pair for every new communication using the MRTD chip's domain parameters $\mathcal{D}_{ICC}$. The ephemeral public key is $\widetilde{PK_{IFD}}$, the corresponding private key is $\widetilde{SK_{IFD}}$.

Note: It is RECOMMENDED that the MRTD chip validates the ephemeral public key received from the inspection system.

3.1.1.2. Operations

- Generating a shared secret K is denoted by $\mathbf{KA}(SK_{ICC}, \widetilde{PK_{IFD}}, \mathcal{D}_{ICC})$ for the MRTD chip and $\mathbf{KA}(\widetilde{SK_{IFD}}, PK_{ICC}, \mathcal{D}_{ICC})$ for the inspection system.

3.1.2. Signatures

The keys and operations for signatures are described in an algorithm-independent way. A mapping to RSA and ECDSA can be found in Appendix A.2.

3.1.2.1. Keys

- The inspection system has a static authentication key pair. The public key is PK_{IFD}, the corresponding private key is SK_{IFD}.

MRTD Chip (ICC)		Inspection System (IFD)
static key pair: $(SK_{ICC}, PK_{ICC}, \mathcal{D}_{ICC})$		
	$\xrightarrow{PK_{ICC}, \mathcal{D}_{ICC}}$	choose random ephemeral keypair $(\widetilde{SK_{IFD}}, \widetilde{PK_{IFD}}, \mathcal{D}_{ICC})$
	$\xleftarrow{\widetilde{PK_{IFD}}}$	
$K = \mathbf{KA}(SK_{ICC}, \widetilde{PK_{IFD}}, \mathcal{D}_{ICC})$		$K = \mathbf{KA}(\widetilde{SK_{IFD}}, PK_{ICC}, \mathcal{D}_{ICC})$

Figure 3.1.: Chip Authentication

3.1.2.2. Operations

- Signing a message m is denoted by $s = \mathbf{Sign}(SK_{IFD}, m)$.
- Verifying the resulting signature s is denoted by $\mathbf{Verify}(PK_{IFD}, s, m)$.

3.2. Chip Authentication

The Chip Authentication Protocol is an ephemeral-static Diffie-Hellman key agreement protocol that provides secure communication and implicit unilateral authentication of the MRTD chip.

3.2.1. Protocol Specification

The following steps are performed by the inspection system and the MRTD chip, a simplified version is also shown in Figure 3.1:

1. The MRTD chip sends its static Diffie-Hellman public key PK_{ICC}, and the domain parameters $\mathcal{D}_{ICC}$ to the inspection system.

2. The inspection system generates an ephemeral Diffie-Hellman key pair $(\widetilde{SK_{IFD}}, \widetilde{PK_{IFD}}, \mathcal{D}_{ICC})$, and sends the ephemeral public key $\widetilde{PK_{IFD}}$ to the MRTD chip.

3. Both the MRTD chip and the inspection system generate the shared secret

$$K = \mathbf{KA}(SK_{ICC}, \widetilde{PK}_{IFD}, \mathcal{D}_{ICC}) = \mathbf{KA}(\widetilde{SK_{IFD}}, PK_{ICC}, \mathcal{D}_{ICC}).$$

4. The MRTD chip hashes the inspection system's ephemeral public key and stores $H(\widetilde{PK_{IFD}})$.

To verify the authenticity of the PK_{ICC} the inspection system SHALL perform Passive Authentication directly after Chip Authentication.

MRTD Chip (ICC)		Inspection System (IFD)
choose r_{ICC} randomly	$\xrightarrow{r_{ICC}}$	
	$\xleftarrow{s_{IFD}}$	$s_{IFD} =$ $\textbf{Sign}(SK_{IFD},$ $ID_{ICC}\|\|r_{ICC}\|\|H(\widetilde{PK_{IFD}}))$
$\textbf{Verify}(PK_{IFD}, s_{IFD},$ $ID_{ICC}\|\|r_{ICC}\|\|H(\widetilde{PK_{IFD}}))$		

Figure 3.2.: Terminal Authentication

3.2.2. Security Status

After a successful key agreement all further communication MUST be protected by Secure Messaging in Encrypt-then-Authenticate mode using session keys derived from K.

Note: The genuineness of the MRTD chip is **implicitly verified** by its ability to perform Secure Messaging using the new session keys. This is accomplished by Passive Authentication as described above.

3.3. Terminal Authentication

The Terminal Authentication Protocol is a two move challenge-response protocol that provides explicit unilateral authentication of the inspection system.

3.3.1. Protocol Specification

The following steps are performed by the inspection system and the MRTD chip, a simplified version is also shown in Figure 3.2:

1. The inspection system sends a certificate chain to the MRTD chip. The chain starts with a certificate verifiable with the CVCA public key stored on the chip and ends with the inspection system's IS Certificate.

2. The MRTD chip verifies the certificates and extracts the inspection system's public key PK_{IFD}. Then it sends the challenge r_{ICC} to the inspection system.

3. The inspection system responds with the signature

$$s_{IFD} = \textbf{Sign}(SK_{IFD}, ID_{ICC}||r_{ICC}||H(\widetilde{PK_{IFD}})).$$

4. The MRTD chip checks that

$$\textbf{Verify}(PK_{IFD}, s_{IFD}, ID_{ICC}||r_{ICC}||H(\widetilde{PK_{IFD}})) = \text{true}.$$

In this protocol ID_{ICC} is the MRTD chip's Passport Number as contained in the MRZ and $H(\widetilde{PK_{IFD}})$ is the hash value of the inspection system's ephemeral Diffie-Hellman public key from the Chip Authentication Protocol.

Note: All messages MUST be transmitted with secure messaging in Encrypt-then-Authenticate mode using session keys derived from the Chip Authentication Protocol.

3.3.2. Security Status

After a successful authentication of the inspection system, the MRTD chip grants access to the stored sensitive data according to the effective authorization level of the authenticated inspection system.

4. Security & Privacy

In this section the formal correctness of the protocols is shown. Following the ideas proposed in [1] a transition from the *Authenticated Link Model* to the *Unauthenticated Link Model* is used to prove the security of the protocols.

Authenticated Link Model: The Authenticated Link Model is an idealized setting where all messages are a priori authenticated.

Unauthenticated Link Model: The Unauthenticated Link Model is the real-world setting where messages are unauthenticated.

The Authenticated Link Model restricts the adversary to attacks on the cryptographic primitive itself and to attacks that do not have impact on the security of the protocol (e.g. denial of service attacks). In this model key agreement would be sufficient to set up the secure channel. The security of the underlying Diffie-Hellman protocol is directly based on the assumption that the Computational Diffie-Hellman Problem is hard.

The transition from the Unauthenticated Link Model to the Authenticated Link Model is done by applying appropriate *Authenticators*, turning unauthenticated messages into authenticated messages. Actually, chip authentication and terminal authentication are such authenticators. Unfortunately, there is no clear definition of the properties of an authenticator in the literature and the corresponding security proofs are quite blurred. To make such proofs more transparent, we give a definition of an authenticator:

Authenticator: A message sent from an originator to a recipient shall be authenticated. It directly follows that the following three properties are sufficient for authentication of the message:

- **Origin:** The recipient must be able to identify the sender of the message.
- **Destination:** The originator must be able to indicate the intended recipient of the message.
- **Freshness:** The recipient must be able to check that the message is not a copy of a previous message.

4.1. Chip Authentication

Chip authentication is similar to the cipher-based authenticator proposed in [1], also shown in Figure 4.1. It is a two move protocol that is used to protect the message m_{ICC} sent from the chip to the terminal by authenticating the message with a MAC.

Chip (ICC)		Terminal (IFD)
	$\xleftarrow{e_{IFD}}$	choose K randomly $e_{IFD} = \mathbf{E}(PK_{ICC}, K)$
$s_{ICC} = \mathbf{MAC}(K, m_{ICC} \| ID_{IFD})$	$\xrightarrow{m_{ICC}, s_{ICC}}$	

Figure 4.1.: Cipher-based Authenticator

To make the transition resulting from the application of the cipher-based authenticator to the basic Diffie-Hellman protocol more clear, consider that the encryption $e_{IFD} = \mathbf{E}(PK_{ICC}, K)$ can be safely replaced by the ephemeral key $\widetilde{PK_{IFD}}$, because this is actually an encryption of $K = \mathbf{KA}(\widetilde{SK_{IFD}}, PK_{ICC}, \mathcal{D}_{ICC})$ (see also Proposition 5 and Remark 1 in [1]).

- **Origin:** Computation of the MAC requires knowledge of the authentication key K. It directly follows from the Computational Diffie-Hellman assumption that only the chip (and the terminal) can generate K from $\widetilde{PK_{IFD}}$.
- **Destination:** The chip includes the identity of the terminal in the MAC. If the terminal remains anonymous, the distinguishing identifier ID_{IFD} can be removed from the MAC. In this case the message is intended for the terminal that is able to verify the MAC (and thus has knowledge of K).
- **Freshness:** If the the terminal chooses the ephemeral key pair $(\widetilde{SK_{IFD}}, \widetilde{PK_{IFD}})$ randomly and uniformly, the authentication key K is also generated randomly and uniformly.

4.1.1. Summarized Properties

Chip Authentication has the following properties:

1. Implicit authentication of the MRTD chip.
2. Secure messaging with forward secrecy.[1]

4.1.2. Remaining Risks

Chip Authentication alone does not necessarily guarantee that the MRTD chip contained in a presented passport is genuine. To preclude sophisticated attacks Chip Authentication MUST be used in combination with Passive Authentication **and** by checking that the printed MRZ and DG 1 of the LDS [5] are equal.

Chip (ICC)		Terminal (IFD)
	$\xleftarrow{m_{IFD}}$	
choose r_{ICC} randomly	$\xrightarrow{r_{ICC}}$	
	$\xleftarrow{s_{IFD}}$	$s_{IFD} =$ $\mathbf{Sign}(SK_{IFD}, ID_{ICC} \| r_{ICC} \| m_{IFD})$

Figure 4.2.: Signature-based Authenticator

4.2. Terminal Authentication

Terminal authentication is the signature-based authenticator proposed in [1], also shown in Figure 4.2. It is a three move protocol that is used to protect the message m_{IFD} sent from the terminal to the chip by authenticating the message with a signature.

- **Origin:** Computation of the signature s_{IFD} requires knowledge of the private key SK_{IFD}. Thus, only the terminal can generate the signature.
- **Destination:** The terminal includes the identity of the chip in the signature.
- **Freshness:** If the chip chooses the challenge randomly and uniformly it is guaranteed that the signature s_{IFD} is recent, as the challenge is included in the signed data.

4.2.1. Summarized Properties

Terminal Authentication has the following properties:

1. Explicit authentication of the inspection system.
2. Key confirmation for secure messaging.

4.2.2. Remaining Risks

Terminal Authentication mitigates the risk introduced by lost or stolen inspection systems by authorizing an inspection system to access sensitive data only for a short period of time. Due to the approximation of the current date, sensitive data may be theoretically read by an already expired inspection system.

On the one hand, an infrequently used passport is obviously more affected by such an attack. On the other hand, the attack is more difficult to mount on infrequently used passports, as access to an MRTD chip still requires consent of the bearer which is enforced by Basic Access Control.

Furthermore, it cannot be prevented that an attacker being able to subvert a terminal gets access to sensitive data.

[1] Assuming that the inspection system chooses $\widetilde{PK_{IFD}}$ randomly and erases the secret key $\widetilde{SK_{IFD}}$ directly after generating the session keys, a compromise of the inspection system's static key pair does not affect the secrecy of past sessions.

4.3. Challenge Semantics

Terminal authentication is a challenge-response protocol based on digital signatures, which is obviously not free from challenge semantics. This is however less important, as the terminal is usually not concerned about its privacy.

Therefore, we only have to show that chip authentication and the cipher-based authenticator provide a non-transferable proof of knowledge. This can be done by showing that the protocol is simulateable without the chip's private key, and that the simulated transcript is indistinguishable from a real transcript. The simulation is trivial:

Input: The chip's static public key PK_{ICC}, the domain parameters $\mathcal{D}_{ICC}$, and a message m_{ICC}.

Output: The authenticated message $s_{ICC} = \mathbf{MAC}(K, m_{ICC}||ID_{IFD})$, where the authentication key is $K = \mathbf{KA}(\widetilde{SK_{IFD}}, PK_{ICC}, \mathcal{D}_{ICC})$ and $\widetilde{SK_{IFD}}$ is a randomly chosen ephemeral private key of the terminal.

In other words, chip authentication is free from challenge semantics because the MAC is based on symmetric cryptography. Any party being able to verify the MAC is also able to compute the MAC.

Appendix A.

Key Management (Normative)

The Object Identifiers used in the following Appendices are contained in the subtree of `bsi-de`:

```
bsi-de OBJECT IDENTIFIER ::= {
  itu-t(0) identified-organization(4) etsi(0)
  reserved(127) etsi-identified-organization(0) 7
}
```

A.1. Chip Authentication Key Pair

A.1.1. Storage on the Chip

The Chip Authentication Key Pair MUST be stored on the MRTD chip.

- The Chip Authentication Private Key is stored in the MRTD chip's secure memory.
- The Chip Authentication Public Key is made available in DG 14 of the LDS [5].[1]

The content of DG 14 is the DER encoded `ChipAuthenticationPublicKeyInfos` specified as follows:

```
ChipAuthenticationPublicKeyInfos ::= SET of
  ChipAuthenticationPublicKeyInfo

ChipAuthenticationPublicKeyInfo ::= SEQUENCE {
  protocol ObjectIdentifier,
  chipAuthenticationPublicKey SubjectPublicKeyInfo,
  keyId INTEGER OPTIONAL
}
```

The data elements contained in a `ChipAuthenticationPublicKeyInfo` data structure have the following meaning:

- The `protocol` identifies the type of protocol to be used with this public key. Valid Object identifiers can be found below.

[1] DG 14 is a reserved data group that was kindly assigned by ICAO for Chip Authentication.

Table A.1.: Algorithms and Formats for Chip Authentication

Algorithm / Format	**DH**	**ECDH**
Key Agreement Algorithm	PKCS#3 [17]	KAEG [8, 10, 3]
Public Key Format	PKCS#3 [17]	ECC [3]
Key Derivation Function	ICAO 3DES KDF [6, 3]	
Ephemeral Public Key Hash	SHA-1 [15]	X-Coordinate

- The `chipAuthenticationPublicKey` contains the public key in encoded form. The specification of `SubjectPublicKeyInfo` can be found in [4].
- The optional `keyId` indicated the local id of the corresponding private key. This field MUST be used, if the private key to be used is not implicitly known to the MRTD chip.

A.1.2. Chip Authentication Object Identifier

The following Object Identifiers are used to identify the algorithm suite (i.e. public key format, key agreement algorithm and key derivation function) for Chip Authentication:

```
id-CA OBJECT IDENTIFIER ::= {
  bsi-de protocols(2) smartcard(2) 1
}

id-CA-DH OBJECT IDENTIFIER   ::= {id-CA 1}
id-CA-ECDH OBJECT IDENTIFIER ::= {id-CA 2}
```

A.1.3. Chip Authentication with DH

For Chip Authentication with DH the respective algorithms and formats from Table A.1 MUST be used. To allow for validation of the inspection system's ephemeral public key the MRTD chip MAY use the mechanism described in [16]. This however requires the MRTD chip to have a more detailed knowledge of the domain parameters, i.e. the order of the subgroup is additionally required.

A.1.4. Chip Authentication with ECDH

For Chip Authentication with ECDH the respective algorithms and formats from Table A.1 MUST be used. Domain parameters MUST be described explicitly in the parameters of the Chip Authentication Public Key. Public keys MUST be in uncompressed format.

A.2. Terminal Authentication Key Pair

A.2.1. Storage on the Chip

For each Terminal Authentication Public Key permanently or temporarily stored on the MRTD chip the following additional data MUST be stored[2]:

- The effective role (i.e. CVCA, DV, or IS) and the effective authorization of the holder of the corresponding private key (cf. Appendix A.3.4).
- The certificate effective date.
- The certificate expiration date.

Furthermore, the MRTD chip MUST make the names of trusted CVCA public keys available to inspection systems in a file EF.CVCA. This file SHALL have FID 0x011C and SFID 0x1C. It SHALL contain a sequence of Certification Authority Reference data objects (cf. Appendix A.3.3.4):

- It SHOULD contain at most two Certification Authority Reference data objects.
- The most recent Certification Authority Reference SHALL be the first data object in this list.

A.2.2. Terminal Authentication Object Identifier

The following Object Identifiers are used to identify the algorithm suite (i.e. public key format, signature algorithm and signature format) for Terminal Authentication:

```
id-TA OBJECT IDENTIFIER ::= {
  bsi-de protocols(2) smartcard(2) 2
}

id-TA-RSA OBJECT IDENTIFIER           ::= {id-TA 1}
TA-RSA-v1_5-SHA-1 OBJECT IDENTIFIER   ::= {id-TA-RSA 1}
TA-RSA-v1_5-SHA-256 OBJECT IDENTIFIER ::= {id-TA-RSA 2}
TA-RSA-PSS-SHA-1 OBJECT IDENTIFIER    ::= {id-TA-RSA 3}
TA-RSA-PSS-SHA-256 OBJECT IDENTIFIER  ::= {id-TA-RSA 4}

id-TA-ECDSA OBJECT IDENTIFIER         ::= {id-TA 2}
TA-ECDSA-SHA-1 OBJECT IDENTIFIER      ::= {id-TA-ECDSA 1}
TA-ECDSA-SHA-224 OBJECT IDENTIFIER    ::= {id-TA-ECDSA 2}
TA-ECDSA-SHA-256 OBJECT IDENTIFIER    ::= {id-TA-ECDSA 3}
```

Further details on the algorithms and formats are specified in the following sections.

[2]The format of the stored data is operating system dependant and out of the scope of this specification.

Table A.2.: Object Identifiers for Terminal Authentication with RSA

OID	Signature Algorithm	Hash Algorithm	Parameters
TA-RSA-v1_5-SHA-1	RSASSA-PKCS1-v1_5	SHA-1	N/A
TA-RSA-v1_5-SHA-256	RSASSA-PKCS1-v1_5	SHA-256	N/A
TA-RSA-PSS-SHA-1	RSASSA-PSS	SHA-1	default
TA-RSA-PSS-SHA-256	RSASSA-PSS	SHA-256	default

A.2.3. Terminal Authentication with RSA

For Terminal Authentication with RSA the following algorithms and formats MUST be used:

Signature Algorithm: RSA [14, 18] as specified in Table A.2.

Public Key Format: The public key consists of three mandatory DER encoded data objects in fixed order (s.a. [13]):

- The respective Object Identifier from Table A.2
- The composite modulus n (Tag 0x81)
- The public exponent e (Tag 0x82)

The length of the modulus n SHALL be at least 1024 bit and a multiple of 512 bit.

A.2.4. Terminal Authentication with ECDSA

For Terminal Authentication with ECDSA the following algorithms and formats MUST be used:

Signature Algorithm: ECDSA with plain signature format [8, 9, 3] as specified in Table A.3.

Public Key Format: The public key consists of two mandatory DER encoded data objects and six optional domain parameters[3] in fixed order (s.a. [13, 3]):

- The respective Object Identifier from Table A.3
- The prime modulus p (OPTIONAL, Tag 0x81)
- The first coefficient a (OPTIONAL, Tag 0x82)
- The second coefficient b (OPTIONAL, Tag 0x83)
- The base point G (OPTIONAL, Tag 0x84)
- The order of the base point r (OPTIONAL, Tag 0x85)

[3] The optional domain parameters MUST be all either present or absent. If the domain parameters are omitted, they are assumed to be implicitly known.

Table A.3.: Object Identifiers for Terminal Authentication with ECDSA

OID	**Signature Algorithm**	**Hash Algorithm**
TA-ECDSA-SHA-1	ECDSA	SHA-1
TA-ECDSA-SHA-224	ECDSA	SHA-224
TA-ECDSA-SHA-256	ECDSA	SHA-256

Table A.4.: CV Certificate Profile

Data Object	**Cert**	**Req**
CV Certificate	m	m
Certificate Body	m	m
Certificate Profile Identifier	m	m
Certification Authority Reference	m	x
Public Key	m	m
Certificate Holder Reference	m	m
Certificate Holder Authorization	m	x
Certificate Effective Date	m	x
Certificate Expiration Date	m	x
Signature	m	m

m: mandatory, x: must not be used

- The public point Y (REQUIRED, Tag 0x86)
- The cofactor f (OPTIONAL, Tag 0x87)

Domain parameters SHALL be taken from [3].

A.3. Certificates and Requests

A.3.1. CV Certificate Profile

The certificates C_{CVCA}, C_{DV} and C_{IS} are self-descriptive Card-Verifiable Certificates (CV certificates). For details on CV certificates see [11, 12, 13]. Those certificates are defined as a sequence of DER encoded data objects (with fixed order) as specified in Table A.4. The signature is created over the **complete** certificate body.

A.3.2. CV Certificate Requests

A CV Certificate Request **Req** is a reduced, self-signed CV certificate. The sequence of data objects (with fixed order) is also specified in Table A.4. The signature is created over the **complete** certificate body.

If a DV applies for a successive certificate, the DV MUST sign the request with the private key of the previous key pair registered with that CVCA. An authentication data object is used to nest the CV Certificate and the additional signature created over the **complete** CV Certificate.

A.3.3. Data Objects

In the following sections the format and encoding of data objects used in CV Certificates is described in more detail.

A.3.3.1. CV Certificate

Tag	0x7F21
Purpose	Nests certificate body and signature
Format	-

A.3.3.2. Certificate Body

Tag	0x7F4E
Purpose	Nests data objects of the certificate body.
Format	-

A.3.3.3. Certificate Profile Identifier

Tag	0x5F29
Purpose	Version of the certificate format, MUST be 0 (Version 1).
Format	Binary encoding in 1 byte.

A.3.3.4. Certification Authority Reference

Tag	0x42
Purpose	Uniquely identifies the issuing CA's signature key pair.
Format	An ISO 8859-1 encoded string of up to 16 characters. It consists of a concatenation of the ISO 3166-1 ALPHA-2 country code of the issuer, the name of the issuer, and the sequence number of the key pair.

A.3.3.5. Public Key

Tag	0x7F49
Purpose	Stores the encoded public key.
Format	The format for RSA and ECDSA public keys is specified in the Appendices A.2.3 and A.2.4, respectively.

A.3.3.6. Certificate Holder Reference

Tag	0x5F20
Purpose	Associates the public key contained in the certificate with a unique name.
Format	An ISO 8859-1 encoded string of up to 16 characters, which consists of a concatenation of the ISO 3166-1 ALPHA-2 country code of the certificate holder, the name of the certificate holder, and the sequence number of the key pair.

A.3.3.7. Certificate Holder Authorization[4]

Tag	0x7F4C
Purpose	Encodes the role of the holder (i.e. CVCA, DV, IS) and assigns read/write access rights to datagroups storing sensitive data
Format	An Object Identifier concatenated with a discretionary data object (cf. Appendix A.3.4).

A.3.3.8. Certificate Effective Date

Tag	0x5F25
Purpose	Date from which on the certificate is valid.
Format	YYMMDD (GMT) encoded as unpacked BCDs.

A.3.3.9. Certificate Expiration Date

Tag	0x5F24
Purpose	Date after which the certificate expires.
Format	YYMMDD (GMT) encoded as unpacked BCDs.

A.3.3.10. Discretionary Data

Tag	0x53
Purpose	Used to store data to determine the authorization of the certificate holder.
Format	Identified by an Object Identifier.

A.3.3.11. Signature

Tag	0x5F37
Purpose	Digital Signature produced by an asymmetric algorithm.
Format	Binary

[4]Note: The tag 0x7F4C is not yet defined by ISO/IEC 7816. The allocation is requested.

Table A.5.: Encoding of Roles and Access Rights

7	6	5	4	3	2	1	0	Description
x	x	-	-	-	-	-	-	**Role**
1	1	-	-	-	-	-	-	CVCA
1	0	-	-	-	-	-	-	DV (domestic)
0	1	-	-	-	-	-	-	DV (foreign)
0	0	-	-	-	-	-	-	IS
-	-	x	x	x	x	x	x	**Access Rights**
-	-	0	0	0	0	-	-	RFU
-	-	-	-	-	-	1	-	Read access to DG 4 (Iris)
-	-	-	-	-	-	-	1	Read access to DG 3 (Fingerprint)

A.3.3.12. Authentication

Tag	0x67
Purpose	Nests a certificate request for a successive certificate with an additional signature.
Format	-

A.3.4. Authorization

The authorization of the holder of the private key corresponding to a certificate is encoded in the Certificate Holder Authorization. The Object Identifier contained in this data object specifies the format and the rules for the evaluation of the authorization level. For Terminal Authentication, the following Object Identifier SHALL be used:

```
id-role-EAC OBJECT IDENTIFIER ::= {
  bsi-de applications(3) mrtd(1) roles(2) 1
}
```

A.3.4.1. Relative Authorization

The *relative authorization* of a certificate holder is encoded in single byte which is to be interpreted as binary bit map as shown in Table A.5. In more detail, this bit map contains a role and access rights. Both are relative to the authorization of all previous certificates in the chain.

A.3.4.2. Effective Authorization

To determine the *effective authorization* of a certificate holder, the MRTD chip MUST calculate a bitwise boolean 'and' of the *relative authorization* contained in the current certificate and effective authorization of the previous certificate in the chain. As the certificate chain always starts with the CVCA public key stored on the MRTD chip, the initial value for the effective authorization is set to the (relative) authorization of the CVCA stored on the chip.

A.3.4.3. Access Rights

The effective authorization is to be interpreted by the MRTD chip as follows:

- The effective role is a CVCA:
  - This link certificate was issued by the national CVCA.
  - The MRTD chip MAY update its internal trustpoint, i.e. the public key and the effective authorization.
  - The certificate issuer is a trusted source of time and the MRTD chip MUST update its current date using the Certificate Effective Date.
  - The MRTD chip MUST NOT grant the CVCA extended access to sensitive data (i.e. the effective access rights SHOULD be ignored).
- The effective role is a DV:
  - The certificate was issued by the national CVCA for an authorized DV.
  - The certificate issuer is a trusted source of time and the MRTD chip MUST update its current date using the Certificate Effective Date.
  - The MRTD chip MUST NOT grant a DV extended access to sensitive data (i.e. the effective access rights SHOULD be ignored).
- The effective role is an IS:
  - The certificate was issued by either a domestic or a foreign DV.
  - If the certificate was issued by a domestic DV, the issuer is a trusted source of time and the MRTD chip MUST update its current date using the Certificate Effective Date.
  - The MRTD chip MUST grant the **authenticated** IS extended access to sensitive data according to the effective access rights.

A.4. CVCA Communication Channels

A robust communication channel is required for all key management tasks (e.g. distribution of new CVCA link certificates and DV Certificate Requests/Responses). Email SHALL be the primary communication channel with a CVCA, however States MAY specify additional online or offline communication channels at their own discretion.

A.4.1. Email

If email is used as communication channel, messages with the following format and where appropriate MIME compliant attachments MUST be used. It is further RECOMMENDED that the sender of a message requests a receipt to ensure that the message was received correctly. If a receipt was requested but no response is received within an appropriate time interval the sender MAY resend the message on the primary or on any secondary communication channel.

Note: Receipts are not signed, and therefore not guaranteed to be authentic.

A.4.1.1. Register

Subject: Register

Body: URIs to be used to contact this state

Attachments: none

A.4.1.2. CVCA Certificate

Subject: CVCA Certificate

Body: Unspecified

Attachments: CVCA Link Certificate(s)

A.4.1.3. DV Certification Request

Subject: DV Certification Request

Body: Unspecified

Attachments: Certificate Request(s)

A.4.1.4. DV Certificate

Subject: [Reply to] DV Certification Request

Body: The reason for not issuing a DV certificate (if a certificate was not issued)

Attachments: DV Certificate(s) (if at least one certificate was issued)

Appendix B.

ISO 7816 Mapping (Normative)

In this Appendix the protocols for Chip Authentication and Terminal Authentication are mapped to ISO 7816 APDUs (Application Program Data Units) [11].

B.1. Chip Authentication

The Chip Authentication Protocol is implemented by the following commands.

B.1.1. MSE:Set KAT

Command

CLA	0x0C	
INS	0x22	Manage Security Environment
P1	0x41	Set for computation
P2	0xA6	Key Agreement Template
Lc	Length of subsequent data field	
Command Data	0x91 0x84	Random number data object containing the encoded $\widetilde{PK_{IFD}}$ (cf. Section A.1). Private key reference data object. This data object is REQUIRED if the private key is ambiguous, i.e. more than one key pair is available for chip authentication.
Le	Absent	

Response

Response Data	Absent	
Status Bytes	0x9000 0x6A80	The key agreement operation was successfully performed. The validation of the ephemeral public key has failed ("Incorrect Parameters in the command data field").

B.1.2. Secure Messaging

Only **after** a successful MSE:Set KAT secure messaging is restarted using the new session keys derived from the key agreement operation, i.e.

- The old session keys and the old SSC are used to protect the response of the MSE:Set KAT command.

- The Send Sequence Counter is set to zero (SSC=0).
- The new session keys and the new SSC are used to protect subsequent commands/responses.

B.2. Terminal Authentication

Terminal Authentication is implemented by the following commands.

B.2.1. MSE:Set DST/AT

Command

CLA	0x0C	
INS	0x22	Manage Security Environment
P1	0x81	Set for verification / external authentication
P2	0xB6 or 0xA4	Digital Signature Template or Authentication Template, respectively.
Lc	Length of subsequent data field	
Command Data	0x83	Control reference data object containing the ISO 8859-1 encoded name of the public key to be set.
Le	Absent	

Response

Response Data	Absent	
Status Bytes	0x9000	The key was selected for the given purpose. Note that some operating systems accept the selection of an unavailable public key and return an error only when the public key is used for the selected purpose.
	0x6A88	The selection failed as the public key is not available (“Referenced data not found”).

B.2.2. PSO: Verify Certificate

Command

CLA	0x0C	
INS	0x2A	Perform Security Operation
P1	0x00	
P2	0xBE	Verify Certificate, self-descriptive
Lc	Length of subsequent data field	
Command Data	Self-descriptive CV certificate to be verified (cf. Section A.3.3.1).	
Le	Absent	

Response

Response Data	Absent	
Status Bytes	0x9000	The certificate was successfully validated and the public was imported.
	other	If the certificate was not accepted, the MRTD chip responds with an operating system dependant error code.

B.2.3. Get Challenge

Command

CLA	0x0C	
INS	0x84	Get Challenge
P1	0x00	
P2	0x00	
Lc	Absent	
Command Data	Absent	
Le	0x08	

Response

Response Data	r_{ICC}	8 bytes of randomness.
Status Bytes	0x9000	Normal operation.

B.2.4. External Authenticate

Command

CLA	0x0C	
INS	0x82	External Authenticate
P1	0x00	Algorithm implicitly known
P2	0x00	Key implicitly known
Lc	Length of subsequent data field	
Command Data	Signature generated by the inspection system.	
Le	Absent	

Response

Response Data	Absent	
Status Bytes	0x9000	Access to datagroups will be granted according to the effective authorization of the corresponding verified certificate.
	other	If the authentication was not successful, the MRTD chip responds with an operating system dependant error code.

B.2.5. Public Key Import

Public keys contained in CVCA link certificates SHALL be **permanently** imported by the MRTD chip. An expired permanently imported public key MAY be overwritten by a subsequent permanently imported public key (cf. Section 2.2.4).

Public key contained in DV and IS certificates SHALL be **temporarily** imported by the MRTD chip. A temporarily imported public key SHALL fulfill the following conditions:

- It MUST remain usable for at least the next cryptographic operation, i.e. PSO: VERIFY CERTIFICATE or EXTERNAL AUTHENTICATE
- It SHALL NOT be selectable or usable after a power down of the MRTD chip.
- It MAY be overwritten by a subsequent temporarily imported public key. An inspection system MUST NOT make use of any temporarily imported public key but the most recently imported.

Further (card OS specific) mechanisms that may be used to handle imported public keys (e.g. MSE: RESTORE) are out of the scope of this specification and SHOULD NOT be used by inspection systems.

B.3. Command Flow

The sequence of ISO 7816 commands required to implement the Advanced Inspection Procedure described in Section 2.1 is illustrated in Figure B.1. In this example the MRZ (DG1), the facial image (DG2), and the fingerprints (DG3) are read from the MRTD chip. It is assumed that the LDS application is already selected and Basic Access Control was successfully performed.

B.4. Extended Length

Depending on the size of the ephemeral Chip Authentication Public Key and/or CV Certificates, APDUs with extended length fields MUST be used to send this data to the MRTD chip. For details on extended length see [11].

While all inspection systems MUST support extended length, MRTD chips NEED NOT support extended length unless the cryptographic algorithms and key sizes selected by the issuing state require the use of extended length.

If the MRTD chip supports extended length this MUST be indicated in the historical bytes of the ATR/ATS or in the EF.ATR as specified in [11].

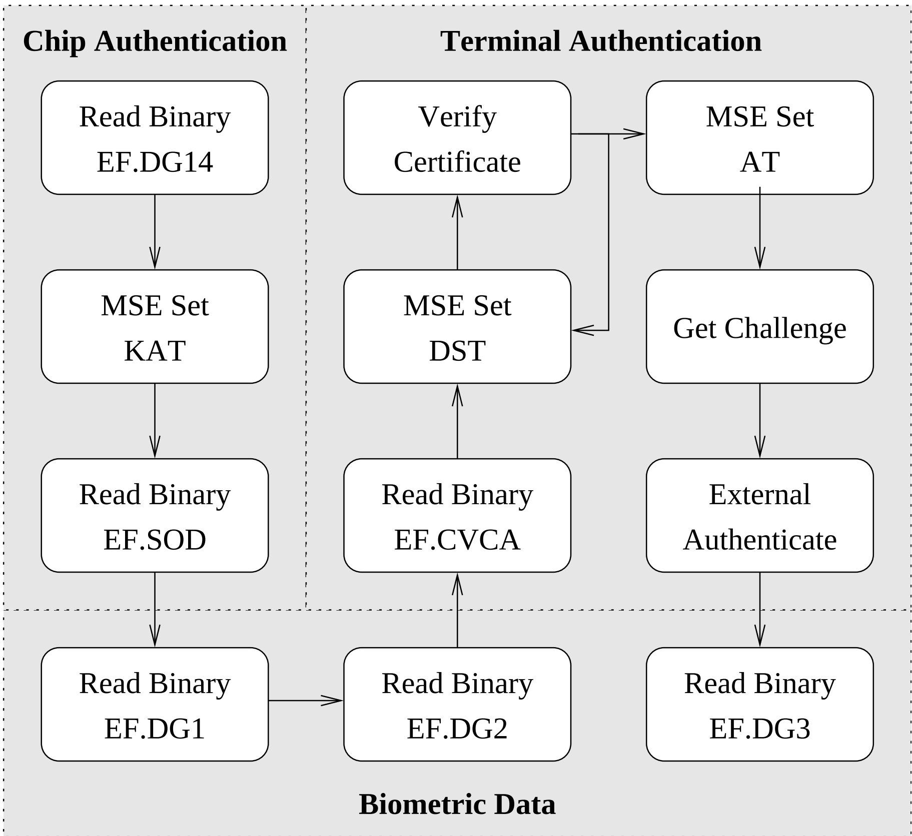

Figure B.1.: Command Flow

Appendix C.

Basic Access Control (Informative)

The protocol for Basic Access Control is specified by ICAO [6]. Basic Access Control checks that the inspection system has *physical* access to the MRTD's data page. This is enforced by requiring the inspection system to derive an authentication key from the *optically* read MRZ of the MRTD. The protocol for Basic Access Control is based on ISO/IEC 11770-2 [7] key establishment mechanism 6. This protocol is also used to generate session keys that are used to protect the confidentially (and integrity) of the transmitted data.

C.1. Document Basic Access Keys

The Document Basic Access Keys K_{ENC} and K_{MAC} stored on the RF-chip in secure memory, have to be derived by the reader from the MRZ of the MRTD prior to accessing the RF-chip. Therefore, the reader optically reads the MRZ and generates the Document Basic Access Keys by applying the ICAO KDF [6] to the most significant 16 bytes of the SHA-1 [15] hash of some fields of the MRZ. As reading the MRZ optically is error-prone, only the fields protected by a check-digit are used to generate the Basic Access Key(s): Document Number, Date of Birth, and Date of Expiry. As a consequence the resulting authentication key has a relatively low entropy. The actual entropy mainly depends on the type of the Document Number. For 10 year valid passports the **maximum** strength of the authentication key is approximately:

- 56 Bit for a numeric Document Number ($365^2 \cdot 10^{12}$ possibilities)
- 73 Bit for an alphanumeric Document Number ($365^2 \cdot 36^9 \cdot 10^3$ possibilities)

Note: Especially in the second case this estimation requires the Document Number to be randomly and uniformly chosen. Depending on the knowledge of the attacker, the actual entropy of the Document Basic Access Key may be lower, e.g. if the attacker knows all Document Numbers in use or is able to correlate Document Numbers and Dates of Expiry.

Given that in the first case the maximum entropy (56 Bit) is relatively low, calculating the authentication key from an eavesdropped session is possible. On the other hand, this still requires more effort than to obtain the same (less-sensitive) data from another source.

MRTD Chip (ICC)		Inspection System (IFD)
		read MRZ optically and derive K
choose r_{ICC} and K_{ICC} randomly		choose r_{IFD} and K_{IFD} randomly
	$\xrightarrow{r_{ICC}}$	
	$\xleftarrow{e_{IFD}}$	$e_{IFD} = \mathbf{E}_K(r_{IFD} \| r_{ICC} \| K_{IFD})$
$r'_{IFD} \| r'_{ICC} \| K'_{IFD} = \mathbf{D}_K(e_{IFD})$		
check $r'_{ICC} = r_{ICC}$		
$e_{ICC} = \mathbf{E}_K(r_{ICC} \| r'_{IFD} \| K_{ICC})$	$\xrightarrow{e_{ICC}}$	
		$r'_{ICC} \| r''_{IFD} \| K'_{ICC} = \mathbf{D}_K(e_{ICC})$
		check $r''_{IFD} = r_{IFD}$

Figure C.1.: Basic Access Control

C.2. Protocol Specification

The Basic Access Control Protocol is shown in Figure C.1. For better readability encryption and message authentication are combined into a single authenticated encryption primitive

$$\mathbf{E}_K(S) = \mathbf{E}'_{K_{Enc}}(S)||\mathbf{MAC}_{K_{MAC}}(\mathbf{E}'_{K_{Enc}}(S)),$$

where $K = \{K_{Enc}, K_{MAC}\}$.

1. The MRTD chip sends the nonce r_{ICC} to the inspection system.

2. The inspection system sends the encrypted challenge $e_{IFD} = \mathbf{E}_K(r_{IFD}||r_{ICC}||K_{IFD})$ to the MRTD chip, where r_{ICC} is the MRTD chip's nonce, r_{IFD} is the inspection system's randomly chosen nonce, and K_{IFD} is keying material for the generation of the session keys.

3. The MRTD chip performs the following actions:

   a) It decrypts the received challenge to $r'_{IFD}||r'_{ICC}||K'_{IFD} = \mathbf{D}_K(e_{IFD})$ and verifies that $r'_{ICC} = r_{ICC}$.
   b) It responds with the encrypted challenge $e_{ICC} = \mathbf{E}_K(r_{ICC}||r'_{IFD}||K_{ICC})$, where r_{ICC} is the MRTD chip's randomly chosen nonce and K_{ICC} is keying material for the generation of the session keys.

4. The inspection system decrypts the encrypted challenge to $r'_{ICC}||r''_{IFD}||K'_{ICC} = \mathbf{D}_K(e_{ICC})$ and verifies that $r''_{IFD} = r_{IFD}$.

After a successful authentication all further communication MUST be protected by Secure Messaging in Encrypt-then-Authenticate mode using session keys derived according to [6] from the common master secret $K_{ICCB} = K_{ICC} \oplus K_{IFD}$.

Note: The keys K_{ICC} and K_{IFD} are different from K_{ICC} and K_{IFD} in the rest of the paper.

Appendix D.

Challenge Semantics (Informative)

Consider a signature based challenge-response protocol between an MRTD chip (ICC) and an inspection system (IFD), where the MRTD chip wants to prove knowledge of its private key SK_{ICC}:

1. The inspection system sends a randomly chosen challenge c to the MRTD chip.
2. The MRTD chip responds with the signature $s = \mathbf{Sign}(SK_{ICC}, c)$.

While this is a very simple and efficient protocol, the MRTD chip in fact signs the message c without knowing the semantic of this message. As signatures are made to be transferable, any third party can be convinced that the MRTD chip has indeed signed this message.

Although c should be a random bit string, the inspection system can as well generate this bit string in an unpredictable but (publicly) verifiable way, e.g. let SK_{IFD} be the inspection system's private key and

$$c = \mathbf{Sign}(SK_{IFD}, A||\text{Date}||\text{Time}||\text{Location}),$$

be the challenge generated by using a signature scheme with message recovery. The signature guarantees that the inspection system has indeed generated this challenge. Due to the transferability of the inspection system's signature, any third party having trust in the inspection system and knowing the corresponding public key PK_{IFD} can check that the challenge was created correctly by verifying this signature. Furthermore, due to the transferability of MRTD chip's signature on the challenge, the third party can conclude that the assertion became true: The MRTD chip was indeed at a certain date and time at a certain location.

On the positive side, countries may use Challenge Semantics for their internal use, e.g. to prove that a certain person indeed has immigrated. On the negative side such proves can be misused to track persons. In particular since Active Authentication is not restricted to authorized readers misuse is possible. The worst scenario would be MRTD chips that provide Active Authentication without Basic Access Control. In this case a very powerful tracking system may be set up by placing secure hardware modules at prominent places. The resulting logs cannot be faked due to the signatures. Basic Access Control diminishes this problem to a certain extent, as interaction with the bearer is required. Nevertheless, the problem remains, but is restricted to places where the passport of the bearer is read anyway, e.g. by airlines, hotels etc.

One might object that especially in a contactless scenario, challenges may be eavesdropped and reused at a different date, time or location and thus render the proof at least unreliable. While eavesdropping challenges is technically possible, the argument is still invalid. By assumption an inspection system is trusted to produce challenges correctly and it can be assumed that it has checked the MRTD chip's identity before starting the Active Authentication Protocol. Thus, the eavesdropped challenge will contain an identity different from the prover's identity who signs the challenge.

Bibliography

[1] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *30th Annual ACM Symposium on the Theory of Computing*, pages 419–428. ACM Press, 1998.

[2] Scott Bradner. Key words for use in RFCs to indicate requirement levels. RFC 2119, 1997.

[3] BSI. Technical Guideline: Elliptic Curve Cryptography (ECC) based on ISO 15946. TR-03111, 2006.

[4] Russel Housley, Tim Polk, Warwick Ford, and David Solo. Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. RFC 3280, 2002.

[5] ICAO NTWG. Development of a logical data structure - LDS for optinal capacity expansion technologies. Technical Report Version 1.7, ICAO, 2004.

[6] ICAO NTWG. PKI for machine readable travel documents offering ICC read-only access. Technical Report Version 1.1, ICAO, 2004.

[7] ISO 11770-2. Information technology – Security techniques – Key management – Part 2: Mechanisms using symmetric techniques, 1996.

[8] ISO 15946-1. Information technology – Security techniques – Cryptographic techniques based on elliptic curves – Part 1: General, 2002.

[9] ISO 15946-2. Information technology – Security techniques – Cryptographic techniques based on elliptic curves – Part 2: Digital signatures, 2002.

[10] ISO 15946-3. Information technology – Security techniques – Cryptographic techniques based on elliptic curves – Part 3: Key establishment, 2002.

[11] ISO/IEC 7816-4:2005. Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange, 2005.

[12] ISO/IEC 7816-6:2004. Identification cards – Integrated circuit cards – Part 6: Interindustry data elements for interchange, 2004.

[13] ISO/IEC 7816-8:2004. Identification cards – Integrated circuit cards – Part 8: Commands for security operations, 2004.

[14] Jakob Jonsson and Burt Kaliski. Public-key cryptography standards (PKCS) #1: RSA cryptography specifications version 2.1. RFC 3447, 2003.

[15] NIST. Secure hash standard. FIPS PUB 180-2 (+ Change Notice to include SHA-224), 2002.

[16] Eric Rescorla. Diffie-Hellman key agreement method. RFC 2631, 1999.

[17] RSA Laboratories. PKCS#3: Diffie-Hellman key-agreement standard. RSA Laboratories Technical Note, 1993.

[18] RSA Laboratories. PKCS#1 v2.1: RSA cryptography standard. RSA Laboratories Technical Note, 2002.